VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Monitorovací systém teplotních profilů pro geotermální aplikace

Monitoring system of temperature profiles for geothermal applications

Bc. Marek Petlák

Diplomová práce

Vedoucí práce: doc. Ing. Michal Prauzek, Ph.D.

Ostrava, 2021

Abstrakt

Práce se zabývá tvorbou systému monitorování teplotních profilů pro geotermální aplikace. Informace z čidla DS18B20 je přes sběrnici 1-Wire zaslána do jednodeskového počítače Raspberry Pi a je zpracována v programu. Takto získané hodnoty teploty jsou zasílány na cloudové řešení Microsoft Azure. Tam se data dále přeposílají do nástroje Power BI, kde je řešena vizualizace naměřených hodnot. V práci je popsán způsob čtení ze senzorů, které jsou připojeny ke sběrnici 1-Wire a mimo jiné také základní principy zasílání dat na Microsoft Azure. Řešení práce může sloužit jako návod pro tvorbu IoT řešení na podobné bázi nebo k využití navrženého přípravku v praxi.

Klíčová slova

IoT, Raspberry Pi, DS18B20, 1-Wire, Microsoft Azure, Power BI, teplota, python, cloud, vizualizace, internet věcí

Abstract

This work deals with the development of a system for monitoring temperature profiles for geothermal applications. The information from the DS18B20 sensor is sent via the 1-Wire bus to the Raspberry Pi computer and is processed in the program. The temperature values obtained in this way are sent to the Microsoft Azure cloud solution. There, the data is further transferred to the Power BI tool, where the measured values are visualized. The work describes the method of reading from sensors that are connected to the 1-Wire bus and among other things, the basic principles of sending data to Microsoft Azure. The solution of the work can serve as a guide for the creation of IoT solutions on a similar basis or to use the proposed product in practice.

Keywords

IoT, Raspberry Pi, DS18B20, 1-Wire, Microsoft Azure, Power BI, temperature, python, cloud, vizualization, internet of things

Poděkování

Rád bych na tomto místě poděkoval panu docentu Michalu Prauzkovi a panu doktoru Jaromíru Konečnému za odborné rady při vypracovávání této diplomové práce.

Obsah

Seznam použitých zkratek a symbolů

A/D převodník	–	Analogově digitální převodník
CRC	–	Cyclic redundancy checks
RPi	–	Raspberry Pi
PC	–	Osobní počítač
FTP	–	File transfer protocol
IDE	–	Integrated development environment
IoT	–	Internet věcí (Internet of Things)
JSON	–	JavaScript Object Notation
Hz	–	Jednotka frekvence (hertz)
Ω	–	Jednotka elektrického odporu (ohm)

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Tato diplomová práce se věnuje tématu internetu věcí (IoT), se kterým se, s rozvojem této oblasti, budeme nejspíše setkávat stále častěji. Čím dál více zařízení je napojeno na internetovou síť. Ať už se jedná o domácí automatizaci, zabezpečení, spotřebiče, chytrá zařízení nebo průmyslovou oblast, IoT síť umožňuje těmto zařízením vyměňovat si data a mít je dostupná odkudkoliv. Zjednodušuje se tak přístup k těmto datům a je umožněna jejich vizualizace či zpracování. Na základě zpracovaných dat pak lze v reálném čase upravovat parametry řídicích aplikací, ovládat zařízení či kontrolovat, zda sledovaný proces probíhá správně.

V práci je popsán jeden ze způsobů a možností využití Raspberry Pi, Microsoft Azure a Power BI v oblasti IoT. Konkrétním tématem práce bylo měření teploty, ale podobné principy a technologie lze aplikovat do nejrůznějších oblastí měření a práce s daty.

Kapitoly 2 až 5 se zabývají teorií k popsanému řešení. Je zde popsán obecný pohled na snímání teploty a technologie, které se k tomu používají. Dále jsou zde popsány vybrané typy senzorů teploty, metody digitalizace a komunikační rozhraní.

V dalších kapitolách je popsáno praktické řešení práce. Postupně je vysvětlen postup práce od senzorické části a čtení teploty až po vizualizaci dat.

Kapitola 2

Senzory teploty pro geotermální aplikace

Teplota je jedna z velmi častých veličin měření. Hodnotu teploty potřebujeme znát z celé řady důvodů. Setkáváme se s ní v meteorologii, fyzice, chemii, potravinářství, geologii, lékařství a mnoha dalších oborech. Hodnota teploty ovlivňuje chemické procesy, vlastnosti, skupenství materiálů atd. Je mnoho způsobů snímání teploty, které pracují na různých principech. Některé z metod jsou popsány v následujících podkapitolách.

2.1 Metody snímání teploty

Teplotu lze měřit například na základě závislosti změny objemu nebo délky materiálu v závislosti na teplotě. Příkladem mohou být teploměry rtuťové (dnes se místo rtuti již používá kvůli bezpečnosti Galinstan. Ternární eutektická slitina gallia, india a cínu), lihové, bimetalové. Plynové teploměry využívají pro svou funkci tepelné roztažnosti plynů.

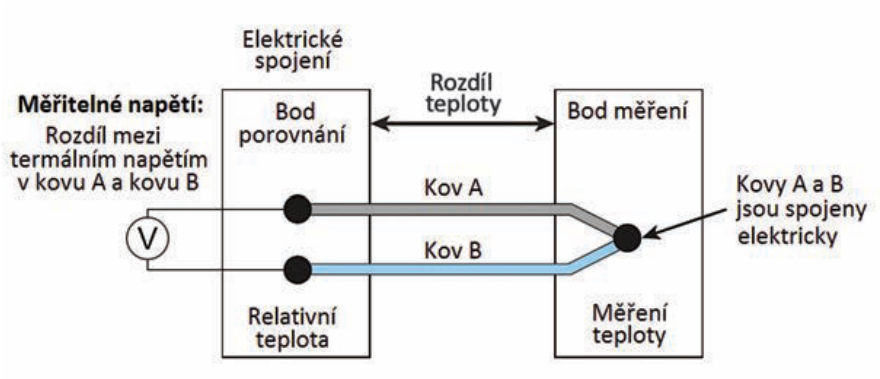
Teplotu lze také měřit na základě viditelného světla vydávaného objektem (těleso vydává záření viditelné okem, jestliže je teplota minimálně 525 °C – Draperův bod) nebo tepelného záření pomocí pyrometru.

Nejpoužívanějšími senzory teploty v elektrotechnice jsou termočlánky, odporové kovové a odporové polovodičové teploměry. [1, 2]

2.1.1 Termočlánky

Termočlánky pracují na principu Seebeckova jevu. Spojením dvou vodičů z různých materiálů v jednom bodě při rozdílu teplot vzniká termoelektrické napětí. Jedná se o přímou přeměnu rozdílu teplot na napětí. Termočlánek je aktivní senzor, jelikož sám generuje napětí a nepotřebuje ke své činnosti napájení. Generované napětí je velmi malé (v řádech mV) s citlivostí v řádu desítek μV na °C v závislosti na použitých materiálech spojení. Schéma termočlánku je zobrazeno na obrázku 2.1.

Navzdory složitějšímu zapojení se tyto senzory používají díky svému velkému rozsahu měřených teplot (-200 až 3 500 °C), malému rozměru sondy a lineární charakteristice. [3]



Obrázek 2.1: Schéma termočlánku [4]

2.1.2 Odporové kovové snímače teploty

Odporové kovové teploměry využívají závislosti odporu kovu na teplotě. Při zvyšující se teplotě roste počet srážek volných elektronů s kladnými ionty v mřížce struktury kovu a tedy roste odpor. Nejčastěji se jako citlivý materiál pro výrobu těchto čidel používá platina díky její chemické netečnosti a možnosti dosáhnout vysoké čistoty materiálu. Mezi dalšími materiály používanými k výrobě odporových kovových čidel je např. nikl, měď či molybden.

Mohou být vyráběny v podobě drátku z daného kovu a nebo s použitím techniky tenkých nebo tlustých vrstev, kdy se kov napaří na korundovou destičku a tak se vytvoří požadovaný rezistor. Drátkové jsou odolnější vůči vibracím a jsou stálejší v čase. Mají však pomalejší odezvu než snímače vyrobené technikou tenkých vrstev. Snímače se dodávají v různých pouzdrech pro zvýšení mechanické odolnosti. Na obrázku 2.2 je snímač Pt100 v pouzdru.

Tyto teploměry se označují např. Pt1000, Ni100 kde číslo, uvedené za označením prvku ze kterého je snímač vyroben, udává základní hodnotu odporu (v Ω) při 0 °C. Typický rozsah senzoru je -200 °C až +1000 °C. [1, 3]

Obrázek 2.2: Odporový teploměr Pt100 [3]

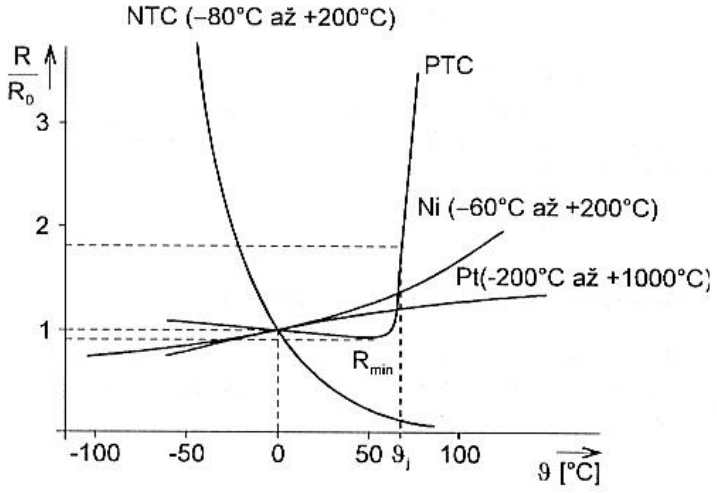
2.1.3 Odporové polovodičové snímače teploty

Odporové polovodičové senzory (termistory) využívají také, jako kovové senzory teploty, vliv teploty na vodivost materiálu. Ta se však u polovodičů chová jinak než u kovových vodičů. Při zvyšující se teplotě roste koncentrace nosičů náboje a tím klesá elektrický odpor senzoru. U klasických polovodičových součástek je tento jev také, avšak způsobem výroby a materiálovým složením se ho snažíme potlačit. U senzorů tohoto jevu naopak využíváme v náš prospěch. Teplotní závislost takových teploměrů má exponenciální charakter. Rozlišujeme dva typy těchto senzorů.

PTC (positive temperature coefficient) – pozistory mají kladný teplotní koeficient, tedy se zvyšující se teplotou roste také jejich odpor podobně jako u kovových senzorů teploty, avšak s jinou charakteristikou s daleko větší citlivostí. Obvyklý rozsah teplot se pohybuje od +60 °C do +180 °C.

NTC (negative temperature coefficient) – negastory mají záporný teplotní koeficient. Se zvyšující se teplotou jejich odpor klesá. Obvyklý rozsah pracovních teplot je od -50 °C do +150 °C.

Navzdory nelinearitě, nižší stabilitě a menšímu teplotnímu rozsahu se termistory používají díky své velké citlivosti na teplotu. Na obrázku 2.3 jsou zobrazeny příklady teplotních charakteristik PTC a NTC termistorů ve srovnání s charakteristikami kovových snímačů teploty (Pt, Ni). [1, 3]



Obrázek 2.3: Charakteristiky termistorů a kovových snímačů teploty [5]

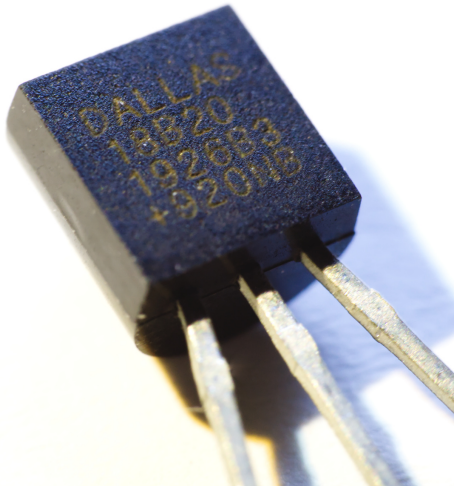
2.2 Vybrané typy teplotních senzorů

Jelikož požadavků pro měření teploty je mnoho, teplotních senzorů existuje celá řada. V následujících podkapitolách jsou popsány 3 vybrané typy.

2.2.1 DS18B20

DS18B20 je digitální teploměr firmy Maxim (dříve Dallas) komunikující po sběrnici 1-Wire (obr. 2.4). Disponuje nastavitelným rozlišením až do 12 bitů. Při 12 bitovém rozlišení je rozlišovací schopnost senzoru 0,0625 °C. Každý senzor má unikátní 64 bitovou adresu dle které je můžeme na sběrnici identifikovat. Připojení na sběrnici 1-Wire umožňuje použití délky vedení v řádech stovek metrů. Délka vyčtení teploty se pohybuje od 94 ms do 750 ms v závislosti od nastaveného rozlišení. Teploměr je dodáván i ve vodotěsném zapouzdření.

Rozsah měření: Od -55 °C do +125 °C. Přesnost ±0,5 °C výrobce uvádí v rozsahu od -10 °C do +85 °C. [6, 7]

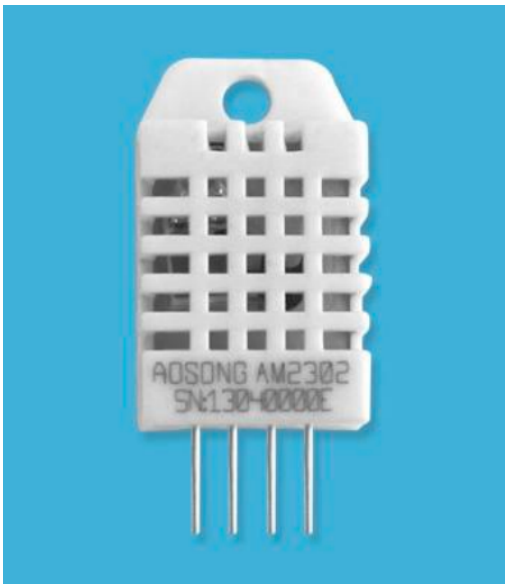


Obrázek 2.4: Senzor DS18B20

2.2.2 AM2302

Digitální senzor teploty a vlhkosti AM2302 od firmy Aosong Electronics komunikuje také po sběrnici 1-Wire (obr. 2.5). Nabízí rozlišovací schopnost 0,1 °C. Pro správné fungování výrobce uvádí nejrychlejší čtení ze senzoru každé 2 s.

Rozsah měření: Od -40 °C do +80 °C. Přesnost ±0,5 °C výrobce uvádí v rozsahu od -20 °C do +80 °C. [8]



Obrázek 2.5: Senzor AM2302 [8]

2.2.3 HTU21D

HTU21D je digitální senzor teploty a vlhkosti od firmy TE Connectivity (obr. 2.6). Komunikuje po sběrnici I^2C. Ve srovnání s předchozími dvěma popsanými senzory má krátkou dobu čtení teploty. Při plném rozlišení (14 bitů) výrobce uvádí maximální dobu čtení 50 ms. Rozlišovací schopnost tohoto senzoru je při nastavení plného rozlišení 0,01 °C.

Rozsah měření: Od -40 °C do +125 °C. Přesnost ±0,3 °C výrobce uvádí v rozsahu od +5 °C do +60 °C. [9]

Obrázek 2.6: Senzor HTU21D [10]

2.3 Teplota v oblasti geotermálních aplikací

V České republice se v oblasti geotermálních aplikací nesetkáváme s vysokými teplotami. Teplota pro potřeby přímého ohřevu vody pro vytápění nebo k přípravě teplé vody (60 – 80 °C) se na území České republiky nachází na některých místech v hloubce 2 km a na většině míst v hloubce 3 km pod povrchem. Kvůli technické náročnosti a obtížné realizaci tak hlubokých vrtů je s tímto zdrojem tepla na našem území prozatím pouze experimentováno.

V posledních letech naopak stále častěji využíváme pro vytápění budov tzv. tepelná čerpadla. Využívá se principu na jakém pracuje např. chladnička či klimatizace. Pro vytápění tepelnými čerpadly je obvykle potřeba vrt 50 až 150 m pod povrch země.

Teplota v hloubkách 100 m pod povrchem se pohybuje kolem 8 – 14 °C. V hloubce 1 kilometru je to pak okolo 25 – 40 °C v závislosti na geografické poloze vrtu. Běžná teplota výstupní vody z tepelných čerpadel je přibližně 55 °C (v případě vysokoteplotních tepelných čerpadel se pohybuje teplota okolo 80 °C) [11, 12, 13]

Kapitola 3

Digitalizace signálů v oblasti teplotních senzorů

Pro zpracování teploty číslicovými metodami potřebujeme informaci o teplotě převést do digitální podoby. Informace se kterou pracujeme na začátku je velikost napětí, která se mění s měnící se teplotou. Z termočlánku ji získáme můstkovým zapojením, z odporových senzorů zase pomocí metody měření odporu, kdy převedeme odpor na velikost napětí. Hodnotu napětí je třeba, pro další zpracování v počítači, převést do digitální podoby. O toto se starají A/D převodníky.

Analogový signál je spojitý v čase a teoreticky obsahuje nekonečné množství informací. Pro převedení analogového signálu na digitální se musí zajistit dva kroky. Vzorkováním se odebere ze signálu konečný počet vzorků a kvantováním jsou vzorky převedeny na digitální hodnoty.

3.1 Vzorkování

Ze spojitého analogového signálu jsou odebrány vzorkováním jednotlivé hodnoty v konkrétních časech. Pro správné vzorkování signálu platí Shannonův–Nyquistův–Kotělnikovův teorém. Podle tohoto teorému musí být pro správné navzorkování signálu (aby byla možná zpětná rekonstrukce) vzorkovací frekvence vyšší než dvojnásobek nejvyšší harmonické frekvence obsažené v signálu, kterou chceme správně vzorkovat. V praxi to znamená, že například audio signál nejčastěji vzorkujeme frekvencí 44 100 Hz. Nejvyšší slyšitelná frekvence se pro lidské ucho uvádí 20 kHz. Tedy výsledná používaná frekvence je podle teorému dvojnásobek + rezerva.

U snímání teploty se frekvence vzorkování pohybují v řádově nižších hodnotách, jelikož samotné senzory mají poměrně dlouhou dobu odezvy. Dokonce se nemusí vzorkovat pravidelně, ale teplota se zjišťuje pouze na základě požadavku.

3.2 Kvantování

Jelikož binárním číslem uloženým v počítači lze vyjádřit čísla s omezenou přesností, je třeba každý vzorek kvantovat. Tedy hodnotě daného vzorku je přiřazeno konkrétní vyjádření binárním číslem. Počet kvantizačních hodnot je odvozen od toho, kolika bitový A/D převodník je použit. Při n bitovém převodníku je to 2^n kvantizačních úrovní. Jestliže je tedy 12 bitový A/D převodník, získává se tak 4 096 úrovní napětí, kterými se může vyjádřit daný vzorek.

Protože se musí jakákoliv hodnota vzorku vyjádřit nějakou z těchto úrovní, hodnoty jsou zaokrouhleny na tu kvantizační úroveň, která je kvantované hodnotě blíže. Takto vzniká kvantizační chyba, která je rovna rozsahu jedné kvantizační úrovně (-1/2 – +1/2 kvantizační úrovně).

U vysokofrekvenčních aplikací je zásadní rychlost A/D převodníku, aby stihl převést dostatečné množství vzorků za vteřinu. Pro potřebu snímání teploty však není třeba dosahovat vysokých rychlostí vzorkování a tedy nároky na rychlost A/D převodu nejsou veliké.

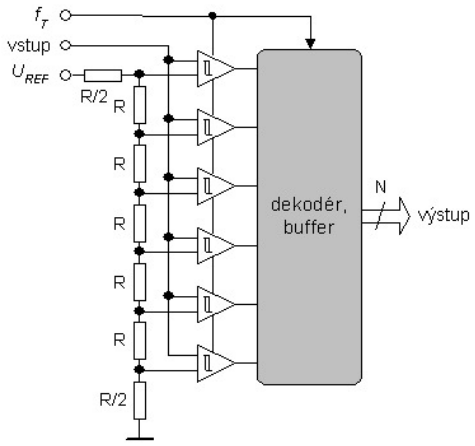
3.3 Druhy A/D převodníků

Stejně jako oblast použití A/D převodníků je veliká, tak požadavky pro jednotlivé aplikace jsou různé. Proto existuje mnoho metod, jak převést analogový signál na digitální. Některé z nich jsou uvedené v následujících podkapitolách.

3.3.1 Paralelní A/D převodník

Tento A/D převodník využívá paralelního zapojení komparátorů napětí a řady rezistorů (obr. 3.1). Na řadu rezistorů je přivedeno referenční napětí. Každý komparátor tedy paralelně porovná v jednu chvíli vstupní napětí s napětím referenčním, které je díky tvořenému napěťovému děliči na každém komparátoru nižší. Komparátory porovnají referenční hodnotu napětí se vstupním a tuto informaci pošlou do dekodéru. V dekodéru se hodnoty z komparátorů převedou na binární číslo.

Tento A/D převodník nabízí vysokou rychlost převodu, avšak je poměrně drahý na výrobu, jelikož pro n bitový převodník je třeba 2^n-1 komparátorů. [14]



Obrázek 3.1: Schéma paralelního A/D převodníku [14]

3.3.2 A/D převodník s postupnou komparací

Protože u předchozího paralelního A/D převodníku je zapotřebí velké množství komparátorů, vznikl A/D převodník s postupnou komparací. Tento využívá stejný princip jako paralelní A/D převodník, ale je zde použito menšího množství komparátorů. Funguje tak, že např. pro 8 bitový převodník je zde použito převodníku 4 bitového. V první fázi se porovná vstupní signál s komparačním napětím a získají se tak vyšší 4 bity výsledné informace. Poté je zbytkové napětí zesíleno (v případě 8 bitového převodníku 16x) a přivedeno zpět k dalšímu porovnání. Tak jsou získány nižší 4 bity výsledné informace.

Takto je docíleno, při zachování vysoké rychlosti převodu, nižšího počtu potřebných komparátorů v obvodu. Pro 8 bitový převodník zde místo 255 komparátorů (u paralelního A/D převodníku) stačí pouze 15. [14]

3.3.3 A/D převodník s postupnou aproximací

U tohoto typu A/D převodníku je použit pouze jeden komparátor. Funguje tak, že při první iteraci procesu se vygeneruje v aproximačním registru hodnota, kde je nejvyšší bit 1 a ostatní jsou v 0. Tato binární hodnota je následně poslána do D/A převodníku, kde se vygeneruje hodnota komparačního napětí. Toto napětí je porovnáno komparátorem se vstupním napětím. Jestliže je zjišťované vstupní napětí vyšší než referenční napětí, je bit v registru ponechán v log. 1. Pokud je vstupní napětí menší, je bit překlopen do log. 0. Poté je do log. 1 nastaven druhý nejvyšší bit a proces se opakuje až do nejnižšího bitu. Tento převodník potřebuje více obslužných obvodů (registry, paměť, D/A převodník). [14]

Kapitola 4

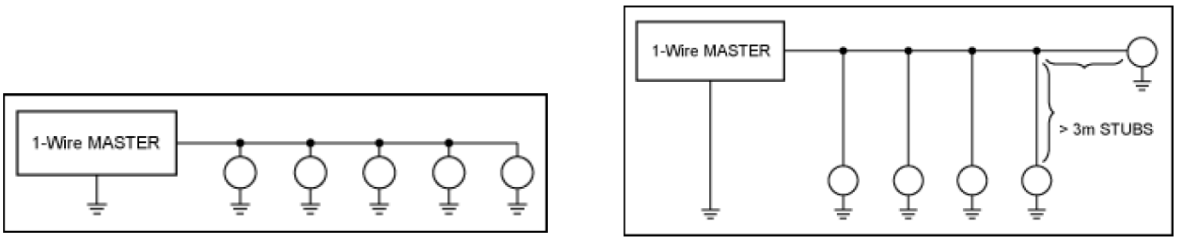
Komunikační rozhraní pro teplotní senzory

Pro komunikaci se senzory a jinými zařízeními existuje mnoho typů sběrnic. Sběrnice slouží ke komunikaci a přenosu dat mezi více zařízeními. Pro úspěšnou realizaci sběrnice je třeba dodržet technické požadavky a komunikační protokol sběrnice. V následujících podkapitolách jsou popsány některé z nich.

4.1 1-Wire

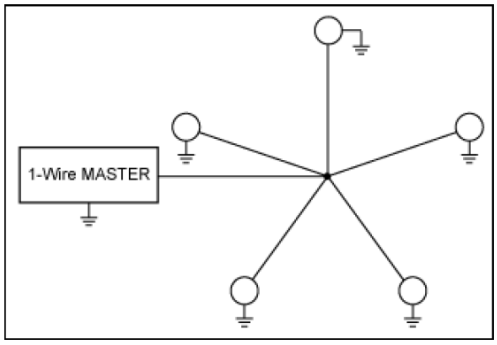
Tato sběrnice byla navržena firmou Dallas Semiconductor. Zařízení je ke sběrnici připojeno pomocí 3 vodičů (dva napájecí a jeden datový vodič) a v případě využití parazitního napájení z datového vodiče lze zařízení připojit pouze pomocí dvou vodičů. Na sběrnici musí být jedno zařízení v režimu „master“, které ovládá jedno nebo více zařízení v režimu „slave“. Všechna zařízení na sběrnici jsou připojena paralelně k datovému vodiči. Základní stav na sběrnici je log. 1 o kterou se stará pull-up rezistor (typicky o hodnotě 4,7 kΩ).

Doporučená topologie pro zapojení zařízení k 1-Wire sběrnici je lineární topologie nebo topologie s odbočkami (delšími než 3 m) vyobrazené na obrázcích 4.1a a 4.1b. Obtížněji realizovatelná je dle specifikace hvězdicová topologie (obr. 4.2), kde mohou nastávat problémy s rušením a komunikací. Délka vedení sběrnice 1-Wire je možná, s dodržením technických požadavků, až do několika stovek metrů. S touto sběrnicí pracují například snímače DS18B20 nebo AM2302 popsané v kapitole 2.2. [7, 15, 16]



(a) Lineární topologie (b) Topologie s odbočkami

Obrázek 4.1: Doporučené topologie 1-Wire sběrnice [7]

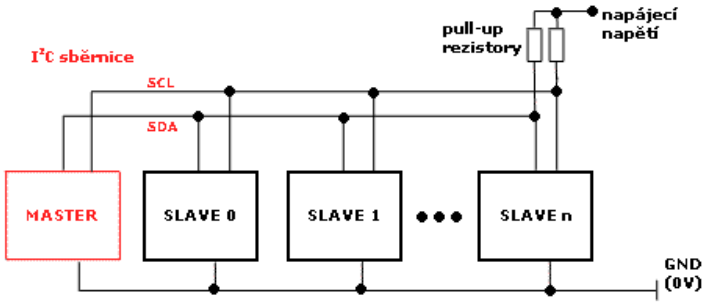


Obrázek 4.2: Hvězdicová topologie 1-Wire [7]

4.2 I²C

Název této sběrnice vznikl ze zkratky IIC bus (Internal-Integrated-Circuit Bus). Vyvinula ji firma Philips. Využívá poloduplexní komunikaci, tedy v jednu chvíli může vysílat pouze jedno zařízení a ostatní mohou naslouchat. Sběrnice obsahuje vodiče SDA (serial data), SCL (serial clock) a společný zemnící vodič (obr. 4.3). Jedná se o synchronní sběrnici. Oba vodiče (SDA a SCL) jsou v základním stavu v log. 1, což je zajištěno pull-up rezistory.

Zařízení „master“ generuje při přenosu na vodiči SCL hodinový signál. Rychlost přenosu musí být přizpůsobena nejpomalejšímu čipu na sběrnici. Všechna zařízení na sběrnici vysílání poslouchají a podle adresy se rozhodne, pro které zařízení jsou informace určeny. Obvyklá délka vedení této sběrnice se pohybuje do vzdálenosti jednotek metrů. Na tuto sběrnici můžeme připojit například senzor HTU21D popsaný v kapitole 2.2. [17, 18]

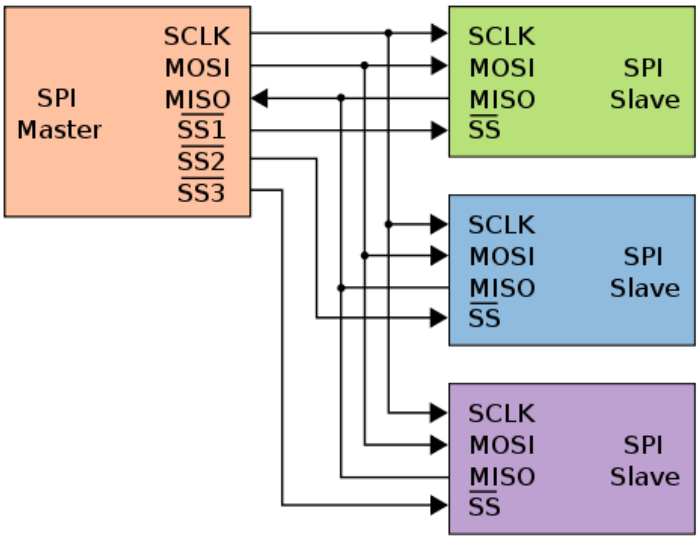


Obrázek 4.3: Zapojení sběrnice I^2C [17]

4.3 SPI

Název sběrnice je zkratkou slov Serial Peripheral Interface. Sběrnice umožňuje synchronní přenos dat mezi jedním zařízením typu „master“ a více zařízeními typu „slave“. Pro komunikaci se používají vodiče „SCLK“ na kterém je generován hodinový signál pro synchronizaci sběrnice. Dále „MISO“ (Master In, Slave Out) a „MOSI“ (Master Out, Slave In) pro samotnou komunikaci. Pro adresaci zařízení se využívá vodiče „SS“ – Slave Select (někdy označovaný jako „CS“ – Chip Select), který je přiveden do každého zařízení zvlášť.

Každý z vodičů sběrnice SPI funguje pouze jednosměrně (slouží buď pro vysílání nebo přijímání dat). Zapojení zařízení ke sběrnici SPI je znázorněno na obrázku 4.4. [19]



Obrázek 4.4: Zapojení sběrnice SPI [20]

Kapitola 5

Přenos dat ze senzorů a cloudová řešení

Pro velké množství dat, která chceme mít přístupná z více míst, je výhodným řešením cloudové úložiště. Je to služba, na kterou jsou data odeslána a uložena na externím serveru. Takto je možné výhodně řešit zálohy uživatelských dat mimo pracoviště, dostupnost totožných dat na více místech pracoviště, home office atd. Lze takto ukládat dokumenty na kterých právě pracujeme, média nebo právě data ze senzorů. Tato data pak lze mít kdekoliv k dispozici a provádět simultánně jejich vizualizaci. [21, 22]

5.1 Přenos dat ze senzorů

Naměřená data ze senzorů se zpracují v programu. Tam se mohou například upravit do požadované přesnosti (zaokrouhlit), provést průměr z několika měření atd. Výsledné hodnoty se pak naformátují do zprávy ve formě JSON (viz výpis 5.1). Tato zpráva je odeslána do konkrétního cloudu pomocí HTTP requestu. Na cloudu tak jsou uložena data a lze je dále zpracovávat v rozhraní služby. [21, 22]

```
# JSON example
{
   "temperature1": 27.71,
   "temperature2": 15.38,
   "temperature3": 21.84
}
```

Listing 5.1: Příklad hodnot teploty ve formátu JSON

5.2 Cloud

Data uložená na cloudu je možné číst, pokud máme k dispozici připojení internetu, odkudkoliv. Tato data si lze stáhnout do počítače a zpracovávat je offline nebo je možnost využití webových aplikací ke zpracování a vizualizaci. Některé cloudové služby nám tyto možnosti nabízejí rovnou ze strany jejich serverů.

Do budoucna se tato technologie bude pravděpodobně dále rozvíjet a rozšiřovat, protože stále více funkcí, které využíváme, pracuje s daty. Data se vyhodnocují a poskytují statistiky či upozorňují na možné problémy (chytré hodinky, prediktivní údržba v průmyslu atd.).

Za využívání cloudu pro ukládání dat nebo jejich zpracování se obvykle platí měsíční poplatky. Výjimku mohou tvořit studentské licence těchto služeb, zkušební období nebo malý objem dat, který nám poskytovatel služby nabízí na vyzkoušení zdarma. [21, 22, 23]

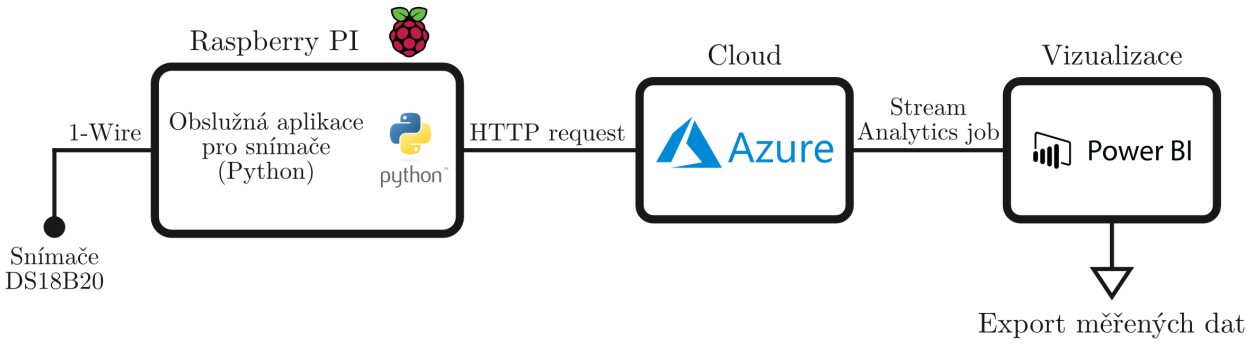
Kapitola 6

Systém pro monitorování teplot

Systém je navržen na bázi jednodeskového počítače Raspberry Pi, snímačů DS18B20 využívající sběrnici 1-Wire, cloudové služby Microsoft Azure a sadou Power BI pro vizualizaci. Obslužná aplikace pro senzory a zasílání hodnot na cloud je psaná v jazyce Python.

6.1 Schéma systému

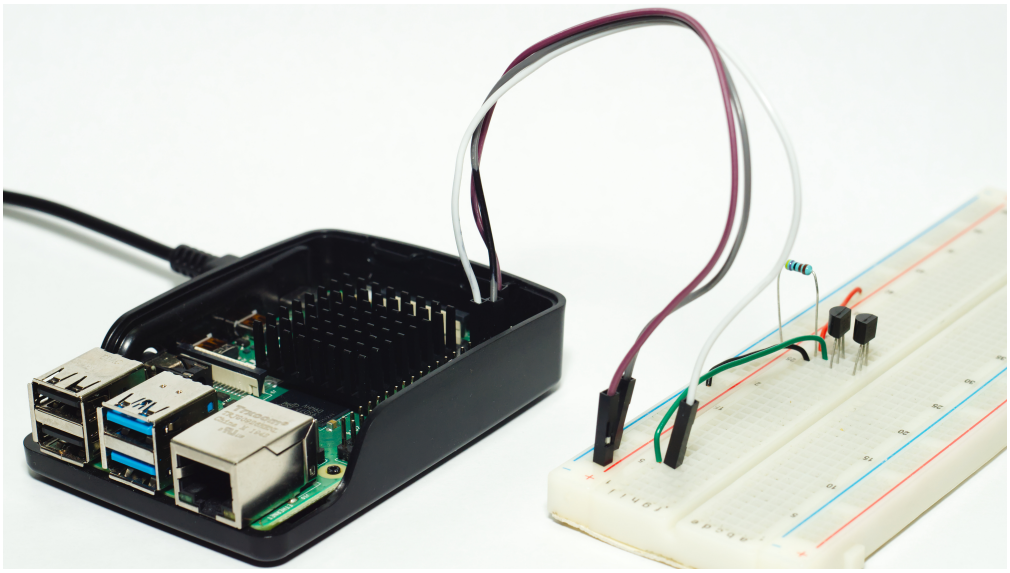
Obrázek 6.1 zobrazuje schéma celého systému pro monitorování teplot. Snímače jsou zapojeny do jednodeskového počítače Raspberry Pi pomocí sběrnice 1-Wire. Na Raspberry Pi je spuštěna aplikace v jazyce Python, která zajišťuje správné vyčtení hodnoty ze snímače na základě adresy. Program přiřadí ke každému snímači název a ve formátu JSON jsou naměřené hodnoty odeslány na cloudovou službu Microsoft Azure. JSON je dále přeposlán do sady Power BI. Tam jsou data vizualizována a je možné provést jejich export.



Obrázek 6.1: Schéma systému pro monitorování teplot

6.2 Hardware

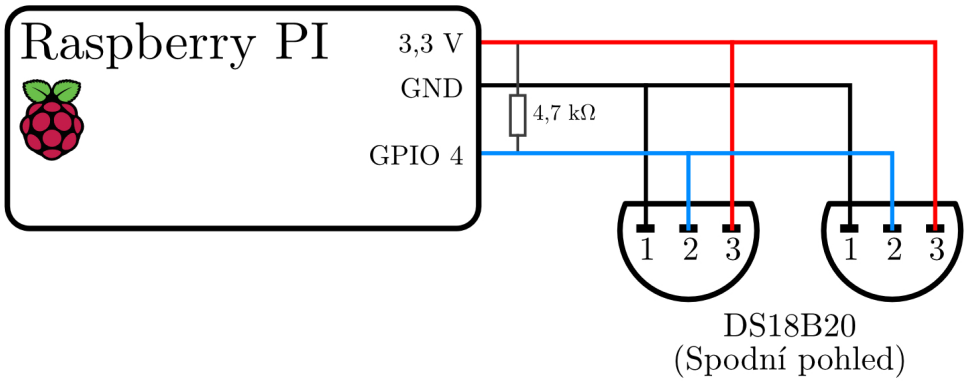
Prototyp je zhotoven na bázi Raspberry Pi 4 Model B. Snímače jsou zapojeny na nepájivém poli, pro možnost rychlé změny počtu snímačů nebo jejich výměny (obrázek 6.2). Tento typ zapojení snímačů odpovídá, dle specifikace výrobce, lineární topologii (viz obrázek 4.1a). Pro účely testování (viz kapitola 10) byly snímače připojeny do nepájivého pole pomocí vodičů pro možnost změny jejich polohy.



Obrázek 6.2: Fotografie zařízení

Na obrázku 6.3 je znázorněno schéma zapojení senzoru DS18B20 k jednodeskovému počítači Raspberry Pi. Piny 3,3 V a GND slouží k napájení snímače. Pin GPIO 4 je využit jako datový vodič pro sběrnici 1-Wire. Rezistor v zapojení o hodnotě 4,7 kΩ slouží jako pull-up rezistor pro datový vodič sběrnice.

Počet senzorů připojených ke sběrnici není teoreticky omezen. Jelikož má každý senzor unikátní adresu, lze je vždy na sběrnici identifikovat. Omezením je především topologie sběrnice, která, při zapojení velkého množství senzorů (řádově stovky), může začít vykazovat chyby nebo přestat fungovat úplně. Takové problémy lze řešit změnou hodnoty pull-up rezistoru nebo úpravou topologie sběrnice.



Obrázek 6.3: Schéma zapojení DS18B20 k Raspberry Pi

6.3 Software

Software systému je možné rozdělit do dvou částí. První část tvoří obslužná aplikace pro senzory, která je blíže popsána v kapitole 7. Tato aplikace zajistí přečtení teplot ze snímačů, jejich zpracování a odeslání na Microsoft Azure. Souběžně probíhá také ukládání dat do textového souboru, který je možné rovněž zpracovávat a archivovat. Tyto soubory jsou ukládány do paměti Raspberry PI.

Druhá část je tvořena využitím služeb Microsoft Azure a Power BI, kterými se zabývají kapitoly 8 a 9. Tato část tvoří cloudové řešení práce, vizualizaci a možnost exportu dat.

Kapitola 7

Aplikace pro obsluhu senzorů

Na jednodeskovém počítači je nainstalován operační systém Raspberry Pi OS (32-bit). Celý systém pro monitorování teplot se skládá ze dvou programů napsaných v jazyce Python.

První program obsahuje třídu a metody pro výpis adresy zapojeného snímače. Pomocí tohoto programu lze zjistit jakou adresu danou výrobcem má každý snímač a na základě této unikátní adresy poté snímače identifikujeme na sběrnici.

Druhý program je hlavní. Zde probíhá cyklické vyčítání teploty, ukládání naměřené teploty do .txt souboru do paměti v zařízení a zaslání dat ve formátu JSON pomocí HTTP requestu na Microsoft Azure.

7.1 Vývoj

Vývoj aplikací na Raspberry Pi byl prováděn pomocí klienta PuTTY, který umožňuje přístup ke konzoli RPi. Dále programu VNC Viewer pomocí kterého byla na PC zobrazována vzdálená plocha RPi. Pro přenos dat mezi PC a RPi byl použit FTP klient FileZilla.

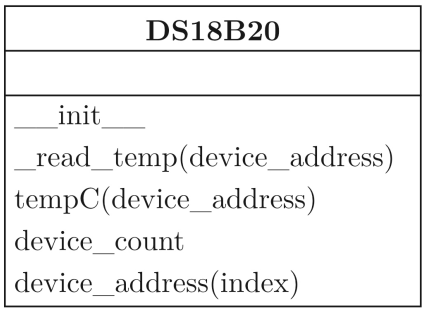
Aplikace pro obsluhu senzorů vznikaly a byly laděny v Python IDE Spyder. Tento program obsahuje mnoho užitečných nástrojů pro psaní kódu v jazyce Python. Umožňuje zobrazení obsahu proměnných, automatické doplňování kódu, zobrazení dokumentace funkcí, nabízí možnost ladění programu a mnoho dalších nástrojů.

7.2 Program pro získání adres

Program **ds18b20.py** (viz příloha I) je určen k získání adres snímačů, roztřídění snímačů a jejich správnému seřazení na sběrnici. Program obsahuje třídu DS18B20 (viz obrázek 7.1), kde jsou definovány metody pro komunikaci se snímači.

7.2.1 Třída DS18B20

Metoda „___init___“ slouží pro inicializaci třídy při vytváření instance. Metoda „_read_temp“ se nevolá přímo. Je volána metodou „tempC“, která slouží pro vyčtení teploty ze snímače na základě zadané adresy. Parametr „device_address“, který obsahuje adresu snímače, se předává do metody „_read_temp“. Pokud je teplota přečtena bez chyb, je návratová hodnota rovna teplotě snímače ve °C. Metoda „device_count“ vrací hodnotu počtu snímačů připojených na sběrnici. Poslední metoda „device_address“ vrací po zavolání adresu senzoru, který je specifikován ze seznamu připojených zařízení na sběrnici 1-Wire parametrem „index“.



Obrázek 7.1: Diagram třídy DS18B20

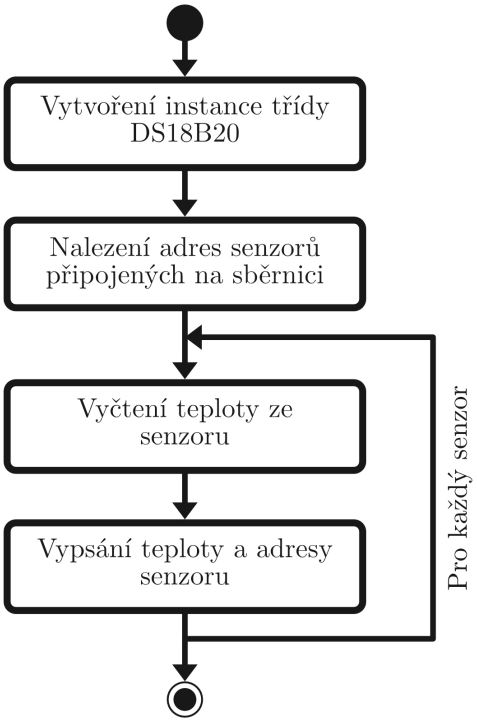
Po spuštění programu se tedy provede následující sekvence. Zavoláním třídy „device_count“ se zjistí, že jsou k 1-Wire sběrnici připojeny například 3 senzory. Pod indexem 0 se nachází adresa prvního senzoru. Na základě tohoto indexu je již možné nechat vypsat adresu senzoru pomocí metody „device_address“ a také zjistit teplotu zavoláním metody „tempC“ s parametrem adresy. V ní se zavolá metoda „_read_temp“. Tato metoda vrací data ze snímače, která byla vyčtena na základě adresy.

V této metodě je zkontrolována CRC (cyklická redundantní kontrola). Tím je ověřeno, že data byla ze snímače přenesena správně. Jestliže na tomto místě kontrola selže, program počká 0,1 vteřiny a opět je proveden pokus o vyčtení teploty. Tento proces se může opakovat nejvýše pětkrát. Jestliže se ani poté nepovede přečíst informaci správně, je místo teploty nahrána do proměnné chybová hodnota a program pokračuje ve čtení z dalšího snímače (pokud je na řadě).

7.2.2 Identifikace senzorů

Průběh programu je znázorněn na obrázku 7.2. Po spuštění programu je vytvořena instance třídy DS18B20, je zjištěn počet připojených snímačů na sběrnici a jejich adresy. Tyto informace jsou uloženy do proměnných a na základě indexu je z každého nalezeného snímače vyčtena teplota. Teplota, spolu s adresou, jsou vypsány do konzole. Na základě zjištěných informací je možné snímače identifikovat podle adresy.

Tento skript proběhne po spuštění pouze jednou. Vypíše teplotu a adresu všech nalezených snímačů na sběrnici. Identifikované adresy je třeba zapsat do pole hlavního programu. V něm je již teplota vyčtena na základě zapsaných adres.



Obrázek 7.2: Schéma programu pro získání adres

Na obrázku 7.3 je zobrazena podoba výpisu konzole při spuštění programu pro získání adres při připojení dvou snímačů. Názvy snímačů „0, 1“ zde reprezentují jejich umístění v rámci nalezených zařízení na sběrnici. Teplota je vypsána pro případ testování přesnosti snímače, či pro zjištění, zdali snímač nevrací chybové hodnoty.

Adresu, která je v konzoli vypsána, je možno přímo zkopírovat do hlavního programu, a tak určit konkrétní senzor.

```
snimac  0 24.19
adresa:  011458440baa
---------------
snimac  1 24.38
adresa:  01145850c4aa
---------------
```

Obrázek 7.3: Výpis teploty a adresy snímačů do konzole

7.3 Hlavní program

Hlavní program **main.py** (viz příloha II) je určen k provozu měřicího systému. Do tohoto programu uživatel musí specifikovat adresy jednotlivých snímačů a poté jejich název. Tyto informace jsou zapsány ve dvou polích. Zde je tedy provázán každý snímač s názvem. Dále je třeba uvést interval čtení. Ten udává dobu, jak často bude teplota přečtena ze všech snímačů. Jelikož čtení z jednoho snímače může při nejvyšší nakonfigurované přesnosti podle výrobce trvat až 750 ms (viz kapitola 2.2.1), je při zadání nižšího času, než je skutečná doba běhu jednoho cyklu měřicí smyčky, čas čekání nulový. To znamená, že v takovém případě se vyčítání provádí nejvyšší možnou rychlostí.

Na obrázku 7.4 je zobrazen výpis z konzole hlavního programu. Každý běh cyklu se vypíše datum, čas odběru teploty a hodnota teploty z každého snímače. Dále následuje podoba JSON zasílaného na cloud a potvrzení, že zaslání proběhlo v pořádku. Nakonec je vypsán uplynulý čas běhu cyklu.

```
2021-03-22 21:23:17
snimac:  1 24.25
snimac:  2 24.38
Sending message: {"temperature1": 24.25,"temperature2": 24.38}
Message successfully sent
Elapsed time: 1.837219 seconds
```

Obrázek 7.4: Výpis měřicí smyčky do konzole

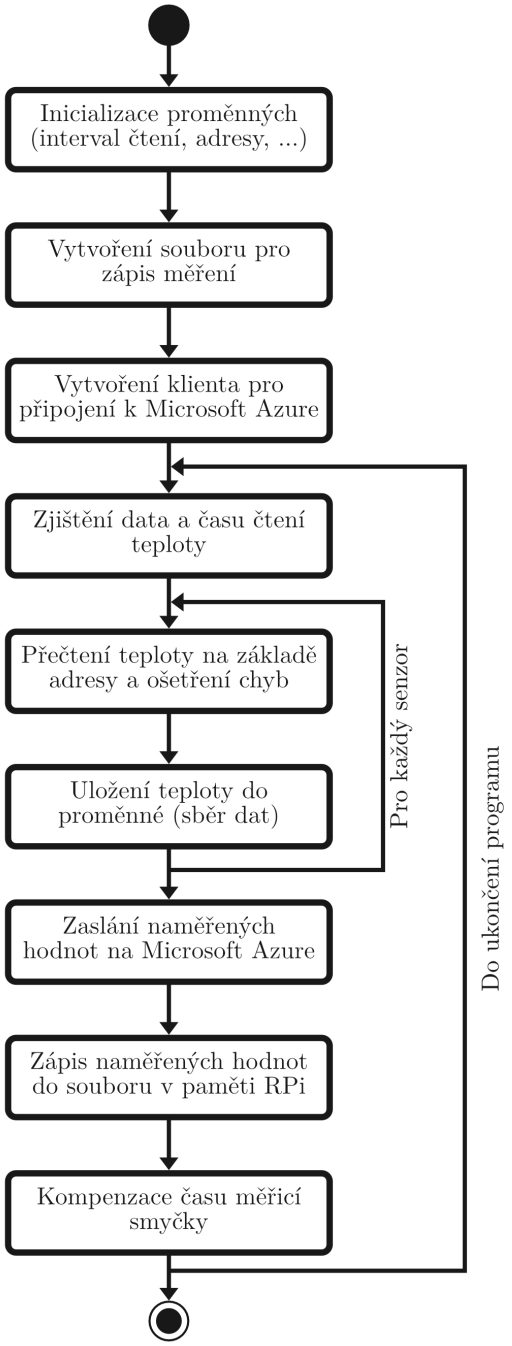
Průběh programu je zobrazen na obrázku 7.5. V první fázi programu jsou naimportovány potřebné knihovny a třída „DS18B20“ popsána v kapitole 7.2. Dále jsou inicializovány proměnné. Mezi nimi se nachází adresy senzorů, interval čtení, connection string pro komunikaci s Microsoft Azure (popsáno v kapitole 7.3.2), cesta k uložení log souboru a další.

Vytvoří se také soubor pro ukládání dat a na první řádek se zapíše hlavička, která obsahuje názvy senzorů a jejich adresy pro možnost zpětné identifikace. Poté je pomocí connection string a příkazu z knihovny Microsoft Azure vytvořen klient ke komunikaci s cloudem.

Nyní se již program dostává do hlavního cyklu, který probíhá opakovaně. Zjistí se aktuální čas, kdy bylo provedeno měření. Na základě zapsaných adres se vyčte teplota ze všech požadovaných snímačů. Proces čtení teploty je stejný jako v programu pro získání adres (kapitola 7.2). Po přečtení všech senzorů se hodnoty naformátují do JSON a odešlou na Microsoft Azure. Stejná data se také, pro archivaci, uloží do log souboru v paměti RPi.

Po celou dobu běhu měřicí smyčky se měří čas jejího trvání. Na konci smyčky proběhne kompenzace času. Aktuální čas trvání smyčky se odečte od požadovaného intervalu čtení. Tento rozdíl je pak doba, po kterou program čeká do dalšího čtení. Opakování tedy probíhá ve stejných časových intervalech nezávisle na tom, zdali čtení bude trvat déle (například kontrola CRC neproběhne v

pořádku). Jestliže je uplynulá doba běhu smyčky delší než požadovaný interval čtení, není možné kompenzovat tuto dobu a program se opakuje ihned.



Obrázek 7.5: Schéma hlavního programu

7.3.1 Senzorická část

Čtení teploty probíhá stejně jako v programu pro získání adres s tím rozdílem, že se čte z konkrétních senzorů na základě známých adres. Není tedy třeba zjišťovat počet senzorů připojených na sběrnici.

Jelikož čtení ze snímačů nemůže probíhat současně, nelze teploty naměřit ve stejnou chvíli. Jestliže je ale interval opakování měření větší než doba vyčítání senzorů a měřené teplotní děje jsou vzhledem k době čtení pomalé, lze na teploty pohlížet, jako by byly přečteny ve stejný čas.

Následující výpis kódu zobrazuje část pro čtení teploty ze senzorů s ošetřením výjimek. Tato sekvence je opakována při každém spuštění měřicí smyčky pro každý senzor. Proměnná „device_address“ vždy obsahuje adresu konkrétního senzoru na sběrnici.

```
try:
   temp = x.tempC(device_address) # čtení teploty ze senzoru DS18B20
   val  = round(temp, 2) # zaokrouhlení hodnoty
   if temp == 998:
      val = "error: CRC invalid"
   if temp == 999:
      val = "error: invalid data"
   print("snimac: ", names[i], val)

except:
   print("error: snímač %s na adrese %s nebyl nalezen" %(names[i], addresses[i]))
   val = "error"
```

Listing 7.1: Sekvence příkazů pro čtení teploty a reakce na chyby

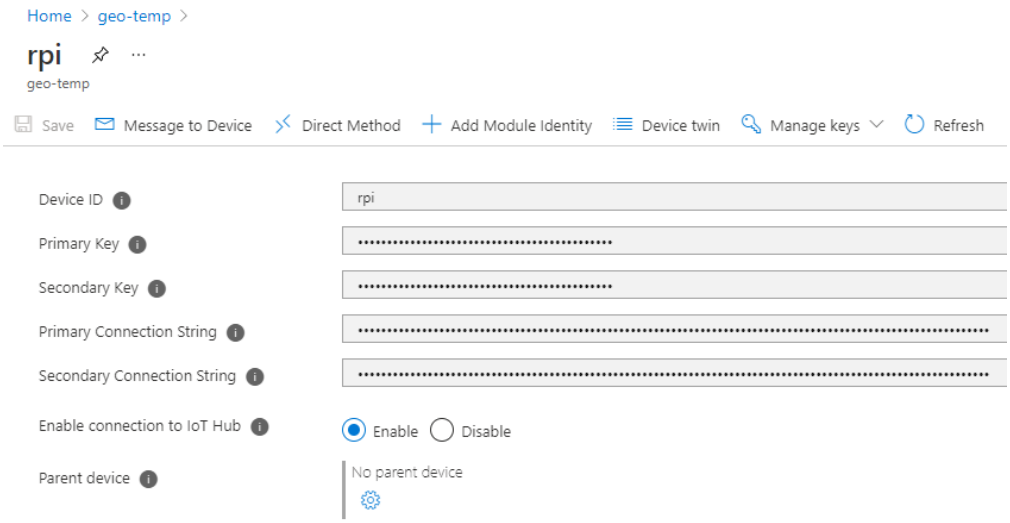
Kontrola CRC probíhá hledáním řetězce „YES“ v řetězci zaslaného senzorem. Řetězec „YES“ znamená, že data byla zaslána správně. Pokud se na tomto místě nachází řetězec „NO“, kontrola CRC neproběhla správně a na základě toho je provedeno opakování měření, v případě pěti neúspěšných pokusů pak ukončení čtení z daného snímače.

Celá sekce čtení ze senzorů je ve struktuře pro zachycení výjimek z důvodu obsloužení neočekávané chyby. V takovém případě se do proměnné s teplotou nahraje řetězec „error“. Do této výjimky se program dostane například při absenci snímače na sběrnici, kdy program nenalezne zařízení s požadovanou adresou. Toto bylo testováno odpojením snímače od sběrnice.

7.3.2 Zasílání dat na Microsoft Azure

Pro zaslání naměřených hodnot na cloud se zašle HTTP request obsahující JSON s hodnotami, které mají být do cloudu zaslány. Je potřeba také „Azure connection string“ řetězec, který slouží k nasměrování dat na správné zařízení v Microsoft Azure. Tento řetězec se vygeneruje po vytvoření

zařízení v IoT Hub (viz. kapitola 8.3). V něm je specifikován název uživatele, název zařízení a samotný klíč pro identifikaci.



Obrázek 7.6: Connection string v Microsoft Azure

Zasílání dat je napsáno za pomoci knihovny „azure-iot-device“ v Pythonu, která poskytuje komunikační rozhraní pro Azure IoT Hub. Lze ji nainstalovat pomocí příkazu „pip install azure-iot-device“.

Pro komunikaci s cloudem je třeba vytvořit instanci IoT Hub klienta, který se vytvoří na základě connection stringu. V tomto případě je v programu uložen v proměnné „CONNECTION_STRING“. Instance se vytváří pouze jednou na začátku programu pomocí následující sekvence příkazů.

```
def iothub_client_init():
client = IoTHubDeviceClient.create_from_connection_string(CONNECTION_STRING)
return client
```

Listing 7.2: Vytvoření IoT Hub klienta

Po vyčtení teplot ze snímačů se informace upraví do formátu JSON. Ten je možno rovnou poslat na cloud za použití následující sekvence příkazů z knihovny azure-iot-device, kde proměnná msg_JSON obsahuje naměřená data ve formátu JSON.

```
message = Message(msg_JSON)
print("Sending message: {}".format(message))

signal.signal(signal.SIGALRM, handler)
signal.alarm(5)
client.send_message(message)
```

```
signal.alarm(0)
print ("Message successfully sent")
```

Listing 7.3: Zaslání HTTP requestu

Je zde také nastaven alarm z knihovny „signal“ který po uplynutí zadané doby (v tomto případě 5 vteřin) zavolá funkci „handler“. V této funkci je vyvolána chyba ve formě TimeoutError. Tato výjimka je zachycena a, kromě výpisu nezdařeného odesílání na Microsoft Azure pro uživatele, je provedeno restartování připojení klienta na Microsoft Azure. Toto slouží pro obsloužení výjimek v podobě dlouhého trvání odesílání na cloud (např. při překročení denního limitu zaslaných zpráv) a jiných neočekávaných událostí. V takovém případě probíhá čtení a ukládání do souboru dále.

```
print("Sending error")
IoTHubDeviceClient.disconnect(client)
client = iothub_client_init()
```

Listing 7.4: Reakce na výjimku při odesílání dat na cloud

Instance „client“, kterou je nutné vytvořit před komunikací s Microsoft Azure, je využívána v běhu programu opakovaně. Ve chvíli, kdy je třeba celou aplikaci ukončit, je nutné odpojit klienta příkazem „IoTHubDeviceClient.disconnect(client)“. Jestliže byla aplikace ukončena bez odhlášení, začne po opětovném spuštění a pokusu navázat nové spojení vykazovat chybové chování.

Proto je příkaz na odpojení klienta umístěn ve všech variantách ukončení programu (přerušení běhu programu pomocí klávesnice, reakce na výjimky).

Kapitola 8

Cloudové řešení Microsoft Azure

Microsoft Azure je cloudová služba, která umožňuje vytváření a konfiguraci vlastních serverů. Jednotlivé služby lze nakoupit, aniž by bylo třeba mít vlastní hardware, který by bylo pro chod takových řešení potřeba. Možnosti použití této služby jsou veliké. Mimo moduly, jakými jsou například IoT Hub nebo SQL server, nabízí Microsoft Azure také pokročilé zpracování dat v podobě strojového učení. Využitím této funkce lze vytvořit moduly, které předpovídají chování a hodnoty zjišťovaných veličin (např. předpověď počasí). Použití v této práci má za účel předvést základní úkony a práci v prostředí Microsoft Azure.

Každá služba má svůj vlastní ceník a je možnost si je nakupovat odděleně, podle toho, co zrovna dané řešení vyžaduje. V případě IoT Hub se ceny liší podle úrovně (Basic, Standard), kde úroveň Standard nabízí více funkcionality (např. zasílání zpráv z cloudu na zařízení nebo využití IoT Edge). Zkušební licence IoT Hub je zdarma k vyzkoušení s omezením na 8 000 zpráv denně. Zprávy se počítají na úrovni IoT Hub. Je to tedy počet zpráv ze všech zařízení (měřicích bodů), které IoT Hub obsahuje. [24]

Tabulka 8.1: Ceník služby IoT Hub úrovně Basic

Edice služby	Měsíční platba (€)	Denní počet zpráv	Velikost zprávy (KB)
B1	8,433	400 000	4
B2	42,165	6 000 000	4
B3	421,650	300 000 000	4

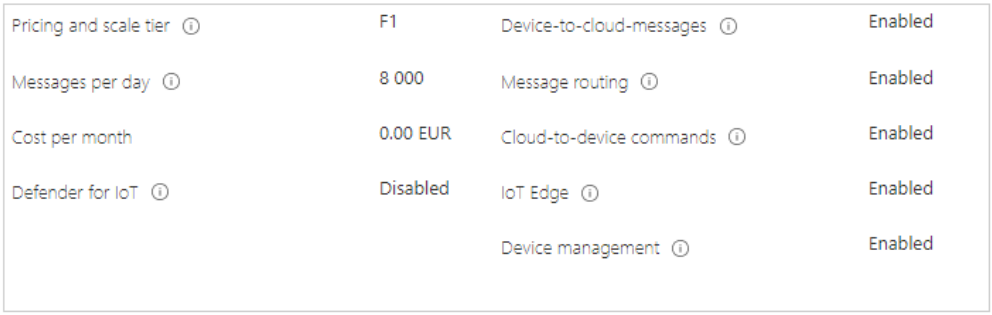
Tabulka 8.2: Ceník služby IoT Hub úrovně Standard

Edice služby	Měsíční platba (€)	Denní počet zpráv	Velikost zprávy (KB)
Free	Zdarma	8 000	0,5
S1	21,083	400 000	4
S2	210,825	6 000 000	4
S3	2 108,25	300 000 000	4

Ceník služby IoT Hub je uveden v tabulkách 8.1 a 8.2. Velikost zprávy je, kromě Free licence, omezena na 4 KB. Jestliže velikost zaslané zprávy překračuje tuto hranici, je zpráva měřena po 4 KB blocích. To znamená, že zpráva o velikosti 10 KB bude počítána jako 3 zprávy. Maximální velikost zprávy, kterou je možno na cloud zaslat, je 64 KB.

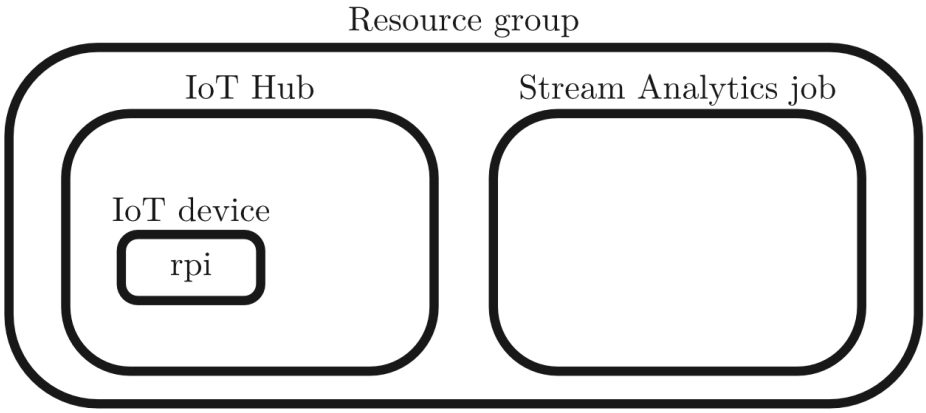
Je tedy třeba provést předběžný návrh řešení se všemi zařízeními a zvážit, která licence bude pro dané použití dostatečná.

Na obrázku 8.1 jsou zobrazeny informace o zkušební licenci IoT Hub. [25]



Obrázek 8.1: Informace zkušební licence IoT Hub

V následujících podkapitolách je uvedeno, jak vytvořit jednotlivé moduly ve službě Microsoft Azure. U každého je popsána konfigurace pro správné fungování. Na obrázku 8.2 je znázorněno blokové schéma tohoto systému.



Obrázek 8.2: Schéma systému v Microsoft Azure

8.1 Resource group

Nejdříve je třeba, aby byla vytvořena skupina prostředků (tzv. resource group). V této skupině se nacházejí všechny prostředky související s daným řešením (viz obrázek 8.2). Pro účely této práce byla založena skupina prostředků s názvem „DP_pet0301“.

Pokud na účtu zatím neexistuje žádná skupina prostředků, vytvoří se nová při vyplňování menu u tvorby modulu (např. IoT Hub v následující kapitole).

8.2 IoT Hub

Při tvorbě IoT Hub zvolíme skupinu prostředků, do které chceme modul přidat. V případě, že ještě žádná na účtu není a nebo je potřeba jiná skupina, je možné vytvořit novou. Pod výběrem skupin se nachází odkaz pro tvorbu nové skupiny. Název musí být unikátní v rámci jednoho účtu.

IoT Hub nazveme jménem, které je unikátní v celém systému (filtr nedovolí použít již rezervovaný název). V další záložce se volí, kterou úroveň IoT Hub chceme používat. V případě úrovně F1 (Free tier), které slouží k vyzkoušení zdarma, je omezení na 8 000 zaslaných zpráv denně (viz obrázek 8.1).

V rámci jedné skupiny prostředků může být vytvořen pouze jeden IoT Hub s úrovní F1. Opět se jedná o omezení, které slouží pouze k vyzkoušení produktu.

8.3 IoT device

Ve chvíli, kdy je vytvořen IoT Hub, lze v jeho rámci vytvářet jednotlivá IoT zařízení. Tato zařízení představují digitální podobu našich měřicích bodů ze kterých jsou odesílána data. V případě

této práce se jedná o jednodeskový počítač Raspberry PI. Při zakládání zařízení je již povoleno volit libovolné jméno, pouze se nesmí shodovat se jménem jiného zařízení v daném IoT Hub. Dále je možnost zvolit si klíč, který poté figuruje v připojovacím řetězci nebo nechat jej automaticky vygenerovat. Pro účely této práce bylo vytvořeno zařízení s názvem „rpi“.

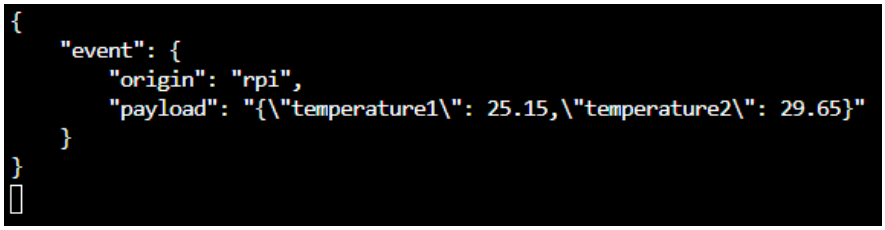
Po vytvoření zařízení můžeme najít v popisu připojovací řetězec, který je potřeba k zasílání zpráv na zařízení v Azure např. z Raspberry PI (obrázek 7.6).

V případě, že je požadováno větší množství měřicích bodů (např. jednotlivých RPi s vlastní řadou teplotních senzorů) se v Microsoft Azure založí více těchto zařízení. Každé bude reprezentovat jeden měřicí bod. Maximální počet zařízení, který je možno vytvořit v jednom IoT Hub, je 1 000 000.

8.4 Zobrazení příchozích zpráv

Pro monitoring příchozích zpráv do Microsoft Azure je nutné do konzole na cloudu přidat rozšíření „azure-cli-iot-ext“ příkazem „az extension add –name azure-cli-iot-ext“. Po otevření konzole ve webovém prohlížeči tedy zadáme tento příkaz. Rozšíření se po chvíli nainstaluje. Poté je možné zadat příkaz pro monitoring příchozích zpráv v IoT Hub „az iot hub monitor-events –hub-name geo-temp –device-id rpi“. V příkazu je specifikován název IoT Hub „geo-temp“ (kapitola 8.2) a název konkrétního zařízení, ze kterého zprávy přichází „rpi“ (kapitola 8.3).

Jestliže jsou na cloud souběžně posílány zprávy, konzole je vypisuje. Na obrázku 8.3 je výstup konzole se zobrazenou JSON zprávou obsahující teploty „temperature1“ a „temperature2“.



Obrázek 8.3: Zobrazená data v konzoli Microsoft Azure

8.5 Consumer group

Před vytvořením Stream Analytics job je třeba přidat „Consumer group“ do IoT Hub. V menu IoT Hub zvolíme záložku „Built-in endpoints“ a tam přidáme název skupiny spotřebitelů (obrázek 8.4). Tento název bude třeba pro nastavení vstupu Stream Analytics job aby bylo možno získávat data z IoT Hub.

Obrázek 8.4: Přidání skupiny spotřebitelů

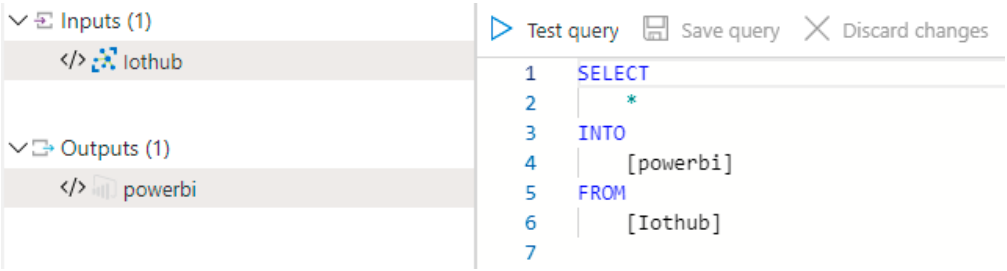
8.6 Stream Analytics job

Pro vytvoření tohoto modulu zvolíme v menu Stream Analytics jobs volbu „Add“. Při tvorbě je opět třeba zadat název, unikátní pouze v rámci modulů ve skupině prostředků, do které má být modul přidán. Pro správnou funkcionalitu je třeba, aby se tento modul nacházel ve stejné skupině prostředků jako zbytek modulů ze kterých mají být data odesílána. To se zajistí volbou v dalším řádku při vytváření, kde se automaticky nabídnou již vytvořené skupiny prostředků. V rámci této práce byl vytvořen modul Stream Analytics job s názvem „Measurement“.

Po vytvoření modulu je nyní třeba zvolit jako vstup pro Stream Analytics job požadovaný IoT Hub. V menu Stream Analytics job je tedy ve volbě „Input“ potřeba zadat konkrétní IoT Hub. Po stisku „Add stream input“ je nutné zvolit IoT Hub, u kterého dále zvolíme alias. S tímto názvem se bude dále pracovat. Poté se zvolí název consumer group (kapitola 8.5). Je zde volba pro formát, ve kterém jsou zprávy zasílány. Automaticky je zvolena možnost JSON, proto je možné nechat ostatní nastavení v původním stavu.

V případě výstupu Stream Analytics job se postupuje podobně. V záložce menu se zvolí „Outputs“. Zde se vybere, v možnosti přidání výstupu, volba Power BI. Po této volbě je třeba se autorizovat účtem Microsoft Power BI. Po autorizaci je možné zadat název výstupu a název datové sady, která je přeposílána do Power BI. Název této datové sady dále figuruje při vizualizaci v Power BI (kapitola 9). Pro účely této práce byla datová sada nazvána „temp“.

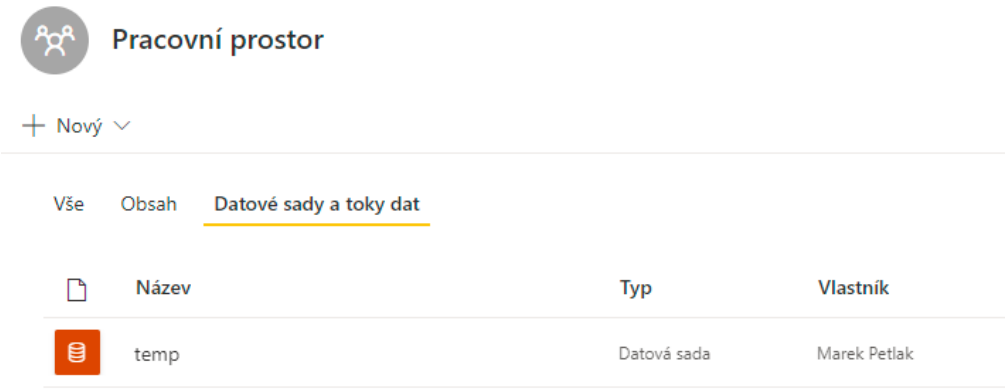
Nakonec je třeba, pro správnou funkci Stream Analytics job, nakonfigurovat dotaz (Query) na základě kterého budou data ze vstupu na výstup předávána. Na obrázku 8.5 je konfigurace taková, že všechny příchozí zprávy na IoT Hub jsou předány do Power BI. Zde se pracuje se zadanými aliasy u tvorby vstupu a výstupu Stream Analytics job. Je možné dosáhnout filtrace zpráv z IoT Hub. Query je například možné nakonfigurovat tak, že bude přeposílat zprávy obsahující pouze konkrétní parametr, který uvedeme.



Obrázek 8.5: Konfigurace query

Zprávy jsou odesílány v rámci IoT Hub. To znamená, že budou odesílány zprávy ze všech zařízení, která jsou vytvořena (kapitola 8.3) a která zasílají zprávy do konkrétního IoT Hub.

Tímto způsobem je provedena správná konfigurace Stream Analytics job pro zasílání dat z IoT Hub do Power BI. Aby tento modul fungoval, je třeba v menu zvolit volbu „Start“, čímž se modul spustí. Pokud jsou na IoT Hub zasílána data a vše je nakonfigurováno správně, po spuštění Stream Analytics job se v Power BI vytvoří datová sada s názvem, který je uveden v konfiguraci výstupu Stream Analytics job (obrázek 8.6).



Obrázek 8.6: Vytvořená datová sada ve webovém rozhraní Power BI

Kapitola 9

Vizualizace Power BI

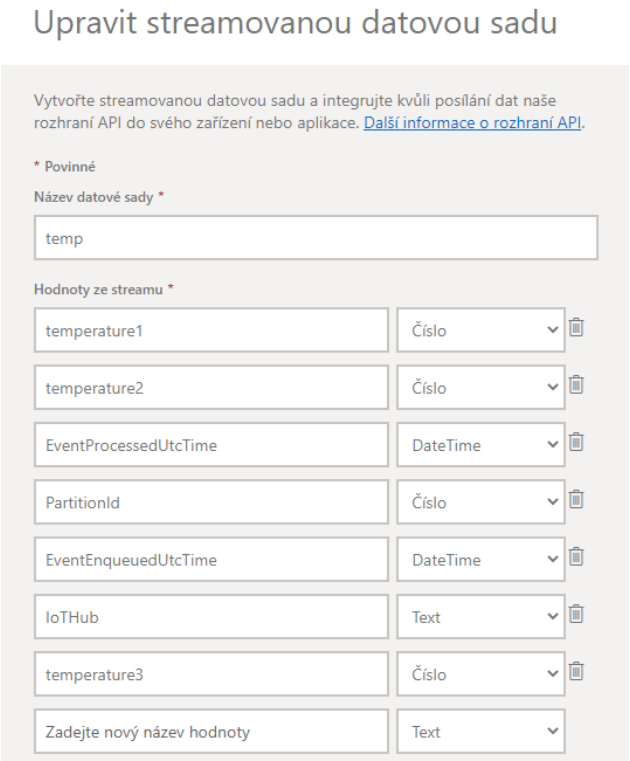
Power BI je platforma pro vizualizaci dat od firmy Microsoft. Nabízí možnosti vizualizovat, analyzovat a sdílet velké objemy dat. Disponuje jednoduchým rozhraním pro koncové uživatele, kteří mají možnost si vytvořit vlastní sestavy a řídicí panely proprietárních aplikací.

V případě této diplomové práce se jedná zejména o vizualizaci teplot z většího počtu senzorů. Forma, jakou se budou hodnoty teploty z každého bodu měření vizualizovat, je tedy na konkrétním uživateli.

9.1 Vizualizace

Pro účely vizualizace slouží datová sada, která je získávána z Microsoft Azure.

Datová sada obsahuje informace o teplotě a čas zpracování požadavku (obrázek 9.1). Tyto hodnoty slouží pro tvorbu vizualizací. V tomto případě je teplota v proměnných začínajících názvem „temperature“. Informaci o čase obsahují proměnné „EventEnqueuedUtcTime“ (čas, kdy byla hodnota přijata) a „EventProcessedUtcTime“ (čas, kdy byla hodnota zpracována). Další proměnná, kterou generuje Stream Analytics job je „Partitionid“. Tato proměnná obsahuje číslo ID oddílu.



Obrázek 9.1: Úprava datové sady v Power BI

```
{
   "temperature1" :98.6,
   "temperature2" :98.6,
   "EventProcessedUtcTime" :"2021-03-06T10:53:08.735Z",
   "PartitionId" :98.6,
   "EventEnqueuedUtcTime" :"2021-03-06T10:53:08.735Z",
   "IoTHub" :"AAAAA555555",
   "temperature3" :98.6
}
```

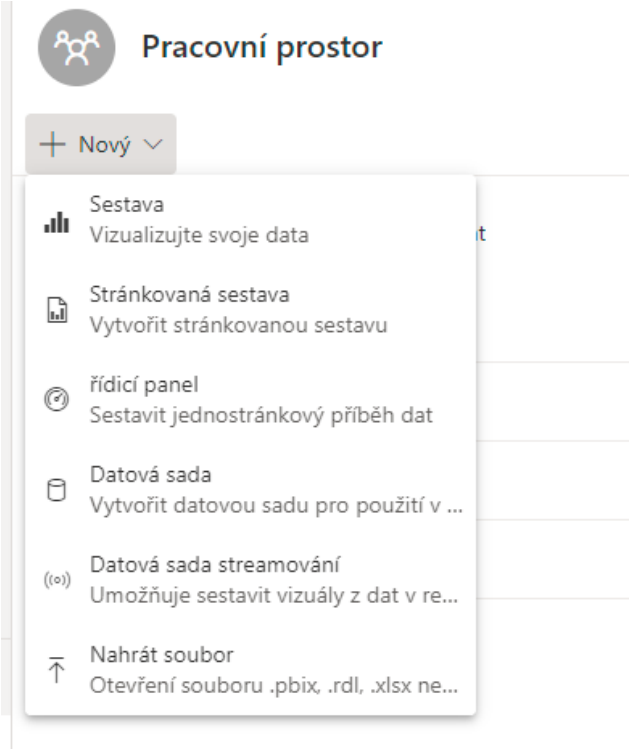
Listing 9.1: Příchozí datová sada ve formátu JSON

V pracovním prostoru Power BI je možné vytvořit řídicí panel, sestavu, stránkovanou sestavu atd. Objekty se liší použitím. V řídicím panelu se vizualizují aktuální data. Při příchozí zprávě s daty se stránka automaticky aktualizuje a tedy všechny grafy a panely obsahují aktuální hodnoty. U sestavy se pracuje zejména s historickými daty. Je možné je zobrazovat v tabulkách či grafech na základě filtrů.

9.1.1 Řídicí panel

Řídicí panel slouží k vizualizaci aktuálních dat. Nenabízí možnosti filtrování nebo volby časového období. Jsou zde k dispozici moduly „Karta“ a „Měřidlo“, které zobrazují poslední příchozí hodnotu. Dále je k vizualizaci možné využít spojnicový, pruhový nebo sloupcový graf. U těchto grafů je volba, za jaké (aktuální) časové období se data zobrazují. Čas se může pohybovat od 1 vteřiny do 60 minut.

Pro založení karty „řídicí panel“ se v záložce menu „pracovní prostor“ zvolí tlačítko „Nový“ a vybere se založení řídicího panelu (viz obrázek 9.2).



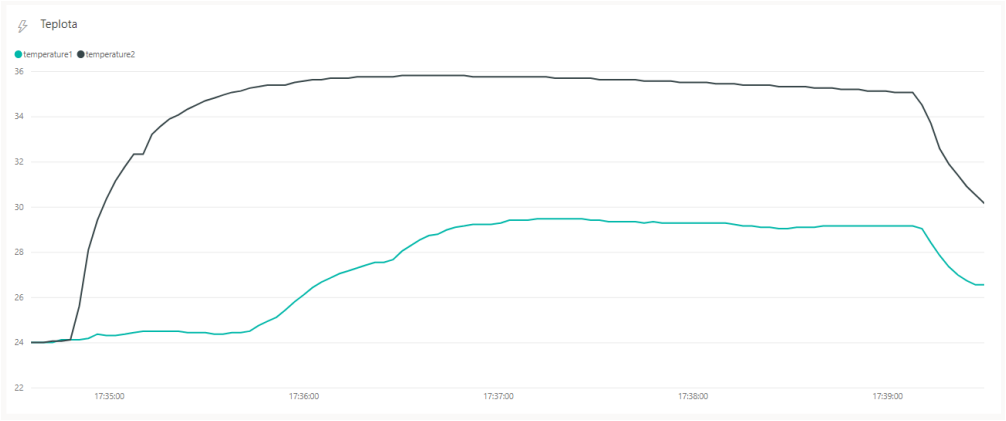
Obrázek 9.2: Vytvoření karty v Power BI

Po vytvoření řídicího panelu je k dispozici prázdná plocha, na kterou lze přidávat tzv. dlaždice. Ty symbolizují jednotlivé objekty, kterým může být obrázek, textové pole, webový obsah,video nebo streamovaná data (data, která chceme vizualizovat). Pro obrázek nebo textové pole lze nastavit funkci objektu, která při kliknutí na objekt přejde na jinou webovou stránku. Tou může být třeba další karta s vizualizací.

Pro vytvoření objektu se tedy zvolí v řídicím panelu „Upravit“ a „Přidat dlaždici“. Při zvolení „Vlastní streamovaná data“ je již možné vybírat mezi způsoby, jak mají být příchozí data z měření vizualizována.

V případě vizualizace pomocí měřidla se zobrazuje pouze aktuální (poslední příchozí) hodnota s ohledem na cílovou hodnotu. Má podobu ručičkového měřidla a lze zvolit minimální, maximální a cílovou hodnotu.

Aktuální hodnotu lze v podobě čísla zobrazit jako kartu. Ta může obsahovat název, podnázev, jednotky a samotnou aktuální hodnotu.



Obrázek 9.3: Graf v Power BI

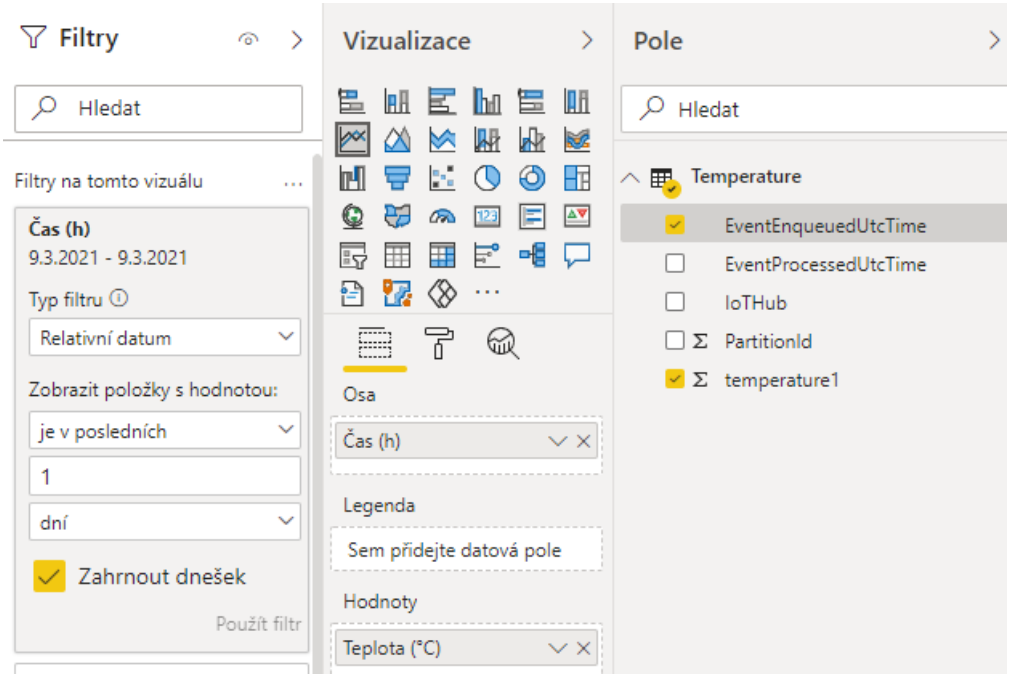
U vizualizace pomocí grafu (viz obrázek 9.3) se vykresluje graf z posledních naměřených hodnot. Časové období pro vykreslení grafu si volí uživatel (od 1 vteřiny do 60 minut). S přibývajícím časem se graf posouvá a staré hodnoty již nejsou viditelné. V nastavení grafu je opět možnost zvolit hodnoty pro časovou osu i osu hodnot. Je možné vizualizovat více průběhů v jednom grafu a tak usnadnit jejich porovnání.

9.1.2 Sestava

Sestava slouží k vizualizaci historických dat. Příchozí hodnoty se v Power BI ukládají a s těmito daty je možné pracovat v sestavě. Sestava se zakládá podobně jako řídicí panel s rozdílem, že již při vytváření se zadává datová sada, ze které budou data zpracovávána. Při zakládání sestavy se tedy jako data zvolí „Vybrat publikovanou datovou sadu“ a v nabídce se již nachází datová sada, se kterou se bude pracovat.

K vizualizaci dat je k dispozici mnoho druhů grafů, měřidel a ukazatelů. Data je možné zobrazit ve formátu matice nebo tabulky.

U všech těchto objektů je třeba zadat, z jakých hodnot bude vizualizace prováděna. V pravé části obrázku 9.4 jsou zadána data pro jednoduchou vizualizaci průběhu teploty na čase.



Obrázek 9.4: Prostředí pro vytváření vizualizace

V levé části obrázku 9.4 je pak vidět filtr pro data, na základě kterých se má graf vykreslit. Je možné vybírat pouze konkrétní hodnoty, relativní datum a čas nebo je k dispozici mnoho jiných pokročilých metod filtrování.

9.2 Export dat

Funkce pro export dat se nachází v možnostech prvku vizualizace. Ty se nacházejí v pravém dolním rohu prvku. Na výběr je export do formátu .csv nebo .xlsx (Excel). Export dat probíhá na základě použitých filtrů. Exportují se tedy ta data, jenž jsou v danou chvíli také vizualizována.

Exportovaný soubor má podobu tabulky s hlavičkou (viz obrázek 9.5). Tato data lze použít pro další vyhodnocování naměřených veličin nebo archivaci.

	A	B	C	D
1	Nejsou použité žádné filtry.			
2				
3	EventProcessedUtcTime	temperature1	temperature2	
4	02.07.21 11:10:07 PM	29,51	36,11	
5	02.07.21 11:10:12 PM	27,99	22,18	
6	02.07.21 11:10:17 PM	26,87	38,42	
7	02.07.21 11:10:23 PM	21,97	31,81	
8	02.07.21 11:10:28 PM	22,21	28,06	
9	02.07.21 11:10:33 PM	30,93	31,97	
10	02.07.21 11:10:38 PM	24,68	24,75	

Obrázek 9.5: Podoba dat v programu Excel

Kapitola 10

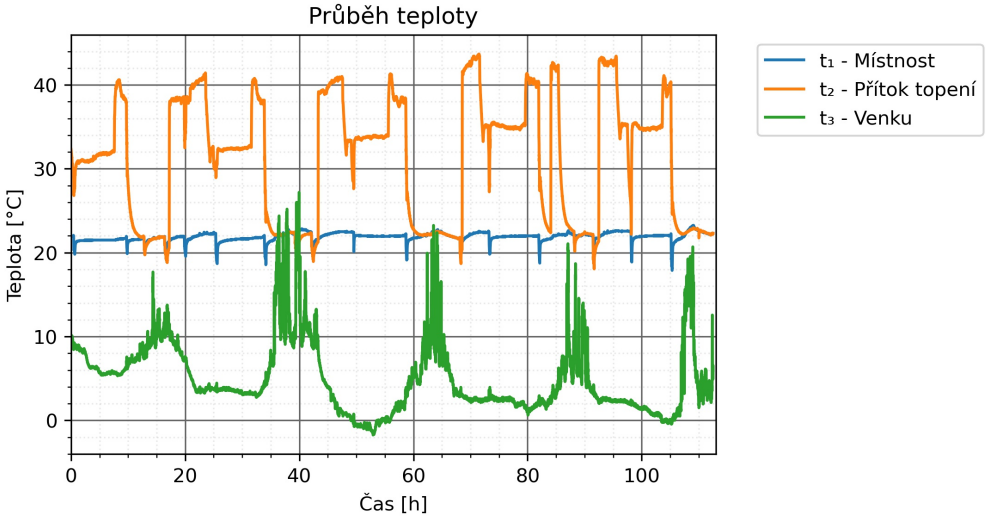
Testování řešení

Pro testování zařízení a měření délky chodu měřicí smyčky byly použity senzory konfigurované na přesnost 12 bitů. Zařízení bylo umístěno v interiéru s výjimkou jednoho senzoru, kterým byla při prvním experimentu čtena venkovní teplota.

10.1 Nepřetržitý chod

Pro tento experiment byly k zařízení připojeny 3 snímače a doba opakování měřicí smyčky byla nastavena na 30 vteřin.

Zařízení bylo spuštěno více než 110 hodin a po celou dobu měření se nevyskytla žádná chyba. Naměřené hodnoty jsou zpracovány do podoby grafu na obrázku 10.1.



Obrázek 10.1: Průběh teploty při testování

Snímače byly v tomto případě zapojeny na vodičích pro snadnější manipulaci a možnost snímat odlišné teploty. Všechny vodiče měly délku kratší než 3 m a tudíž se z hlediska topologie sítě 1-Wire stále jedná o lineární topologii zapojení.

10.2 Trvání měřicí smyčky

Doba běhu smyčky byla měřena od prvního řádku měřicího cyklu do vykonání posledního příkazu cyklu pomocí nadefinované funkce přímo v programu.

Tento čas je také zobrazován do konzole a slouží pro kompenzaci doby opakování měření. Zadaný čas opakování měření je definován na začátku programu. Na konci měřicí smyčky se tento čas porovná s dobou, která uplynula od počátku smyčky a je odečtena od zadané doby.

Hodnoty v tabulce 10.1 byly vybrány náhodně, vždy při daném počtu snímačů. Bylo vybráno 10 hodnot a z nich poté vypočítán průměr.

Tabulka 10.1: Doba běhu měřicí smyčky (s) v závislosti na množství snímačů

	Počet snímačů			
Číslo měření	1	2	3	4
1	0,983	1,837	2,761	3,611
2	0,944	1,842	2,790	3,599
3	1,045	1,839	2,733	3,585
4	0,993	1,850	2,746	3,608
5	0,949	1,837	2,854	3,609
6	1,054	1,834	2,787	3,588
7	1,172	1,927	2,801	3,600
8	1,036	1,914	2,757	3,603
9	0,962	1,848	2,766	3,604
10	0,987	1,829	2,721	3,626
Průměrná doba (s)	1,013	1,856	2,772	3,603

Doba běhu měřicí smyčky se může prodloužit u příkazu pro zaslání dat na Microsoft Azure. Při testování překročení denního limitu počtu zpráv (viz obrázek 8.1) se program na příkazu pro zaslání zastavil a data byla zaslána až následující den (po půlnoci). Proto je na tomto místě nastaven časovač, který po uplynutí určité doby trvání příkazu pro zaslání dat (5s) vyvolá výjimku. V obsluze výjimky se restartuje připojení klienta k Microsoft Azure a měření pokračuje dále. Data sice nebyla zaslána na cloud, ale měření pokračuje a všechny hodnoty se zapisují do .txt souboru na RPi.

Druhé místo, kde je možné, že se program zdrží déle, je při samotném čtení ze senzoru. Zde je, po zaslání dat ze senzoru, zkontrolováno CRC (cyclic redundancy checks). To slouží ke kontrole

správně zaslaných dat. V případě, že tato kontrola neproběhne v pořádku, program vyčká 0,1 s a opakuje čtení ze senzoru. To může proběhnout až pětkrát. Pokud se přenos dat přesto nepovede, je do souboru s hodnotami zapsána chyba a měření pokračuje dále.

10.3 Zhodnocení testování

Za celou dobu testování nenastala žádná neočekávaná chyba.

Hlavní proměnnou složkou u doby trvání měřicí smyčky je odesílání naměřených dat na Microsoft Azure. Doba trvání odeslání je proměnná v řádu desetin vteřiny.

Při odpojení snímače program nenalezne zařízení s danou adresou na sběrnici a je obsloužena výjimka, kde je tato událost vypsána do konzole a dále probíhá čtení dalšího snímače. Po připojení snímače je z něho opět vyčítána teplota.

Vzhledem k proměnlivosti času, který je potřeba pro vykonání příkazů v cyklu, je použitý způsob časování měřicí smyčky (měření času trvání a následné odečtení od požadovaného intervalu čtení) na místě.

Kapitola 11

Závěr

V rámci diplomové práce byl navržen a realizován systém pro měření teplot s použitím senzorů DS18B20, jednodeskového počítače Raspberry Pi, služby Microsoft Azure a nástroje pro vizualizaci Power BI. Byla ukázána jedna z cest, jak lze využít tyto technologie v rámci IoT.

V práci jsou popsány způsoby, jak byly odstraněny nedostatky, které se projevily u použitých technologií (např. překročení denního limitu zpráv na Microsoft Azure nebo ukončení programu bez odpojení klienta pro komunikaci s cloudem).

Jelikož je téma IoT obsáhlé, lze tuto práci dále rozvíjet mnoha směry. V rámci aplikace pro obsluhu senzorů je možné do programu implementovat další funkce zpracování naměřených dat, která se budou odesílat na cloud. Dále by bylo možné z dat, rovnou v rámci aplikace, generovat grafy a zasílat je na další služby (Telegram, Discord aj.). Toto zasílání by se provádělo v určitých intervalech (podle potřeby např. každý den, měsíc) nebo na vyžádání od uživatele prostřednictvím příkazů.

U této konkrétní aplikace tvoří Microsoft Azure pouze mezičlánek v přenosu dat. Tímto byl ukázán úvod do práce s touto službou (zasílání dat a práce v prostředí Microsoft Azure). Z pohledu dalšího rozvoje práce se tedy v této části nabízí využití dalších nástrojů, které Microsoft Azure nabízí (SQL databáze, strojové učení a další).

Literatura

1. SMUTNÝ, Lubomír. *Časopis Automa Snímače teploty – současný stav a směry vývoje* [https://automa.cz/cz/casopis-clanky/snimace-teploty-soucasny-stav-a-smery-vyvoje-2007_05_34041_2168/]. 2007-04. (Accessed on 12/04/2020).
2. MICHALSKI, L.; ECKERSDORF, K.; KUCHARSKI, J.; MCGHEE, J. *Temperature Measurement*. Wiley, 2001. Wiley series in measurement science and technology. ISBN 9780471867791. Dostupné také z: https://books.google.cz/books?id=pP28yxRZ7i8C.
3. VOJÁČEK, Antonín. *Přehled principů el. měření teploty - 1. díl* [https://automatizace.hw.cz/prehled-principu-el-mereni-teploty-1-dil]. 2014-06. (Accessed on 12/04/2020).
4. NGONGANG, Aurelien. *Snímače IR MEMS Thermopile Array* [https://www.dps-az.cz/soucastky/id:26125/z-aktualniho-vydani-casopisu-snimace-nbsp-ir-nbsp-mems-thermopile-array]. 2016-03. (Accessed on 12/06/2020).
5. VOJÁČEK, Antonín. *Teorie zpracování signálu platinových teplotních senzorů | Automatizace.HW.cz* [https://automatizace.hw.cz/zpracovani-signalu-platinovych-senzoru]. 2014-07. (Accessed on 12/04/2020).
6. *DS18B20: Programmable Resolution 1-Wire Digital Thermometer*. Rev 6. Maxim Integrated Products, Inc., 2019.
7. *1-Wire: GUIDELINES FOR RELIABLE LONG LINE 1- WIRE NETWORKS*. 1. vyd. Maxim Integrated Products, Inc., 2014.
8. *AM2302: Temperature and humidity module AM2302 Product Manual*. 1. vyd. Aosong(Guangzhou) Electronics Co.,Ltd., 2020.
9. *HTU21D(F) RH/T SENSOR IC: Digital Relative Humidity sensor with Temperature output*. 1. vyd. TE Connectivity Ltd. family of companies, 2017.
10. LUBOŠ, M. *Senzor teploty a vlhkosti HTU21D I2C* [https://navody.arduino-shop.cz/navody-k-produktum/senzor-teploty-a-vlhkosti-htu21d-i2c.html]. 2017-05. (Accessed on 12/06/2020).

11. ČÍŽEK, Petr. *Jak geologické poměry ovlivňují provoz tepelných čerpadel - TZB-info* [https://www.tzb-info.cz/2190-jak-geologicke-pomery-ovlivnuji-provoz-tepelnych-cerpadel]. 2004-10. (Accessed on 12/04/2020).

12. ŠAFANDA, Jan. *Teplo z nitra Země - Časopis Vesmír* [https://vesmir.cz/cz/casopis/archiv-casopisu/2008/cislo-9/teplo-z-nitra-zeme.html]. 2008-09. (Accessed on 12/06/2020).

13. ENBRA, a.s. *Pozor na parametry a konstrukci tepelného čerpadla - TZB-info* [https://vytapeni.tzb-info.cz/tepelna-cerpadla/10899-pozor-na-parametry-a-konstrukci-tepelneho-cerpadla]. 2014-02. (Accessed on 12/06/2020).

14. PROKEŠ, Aleš. *Vzorkování a A/D převod signálů v radiotechnice* [http://www.elektrorevue.cz/clanky/03046/index.html#k3]. 2003-12. (Accessed on 12/18/2020).

15. MALÝ, Martin. *Sběrnice 1-Wire™* [https://vyvoj.hw.cz/navrh-obvodu/rozhrani/sbernice-1-wiretm.html]. 2004-11. (Accessed on 12/12/2020).

16. STEHLÍK, Petr. *Internetový termostat: stavba sítě teplotních čidel na 1-Wire sběrnici - Root.cz* [https://www.root.cz/clanky/internetovy-termostat-stavba-site-teplotnich-cidel-na-1-wire-sbernici/]. 2015-05. (Accessed on 12/12/2020).

17. OLEJÁR, Martin. *Stručný popis sběrnice I2C a její praktické využití k připojení externí eeprom 24LC256 k mikrokontroléru PIC16F877* [https://vyvoj.hw.cz/navrh-obvodu/strucny-popis-sbernice-i2c-a-jeji-prakticke-vyuziti-k-pripojeni-externi-eeprom-24lc256]. 2000-05. (Accessed on 12/12/2020).

18. TIŠNOVSKÝ, Pavel. *Komunikace po sériové sběrnici I2C - Root.cz* [https://www.root.cz/clanky/komunikace-po-seriove-sbernici-isup2supc/]. 2009-01. (Accessed on 12/13/2020).

19. TIŠNOVSKÝ, Pavel. *Externí sériové sběrnice SPI a I²C - Root.cz* [https://www.root.cz/clanky/externi-seriove-sbernice-spi-a-i2c/]. 2008-12. (Accessed on 12/13/2020).

20. ČÍŽEK, Jakub. *Pojďme programovat elektroniku: Co se skrývá uvnitř běžné SD karty a jak ji oživit – Živě.cz* [https://www.zive.cz/clanky/pojdme-programovat-elektroniku-co-se-skryva-uvnitr-bezne-sd-karty-a-jak-ji-ozivit/sc-3-a-203150/default.aspx]. 2020-03. (Accessed on 12/13/2020).

21. ZHOU, H. *The Internet of Things in the Cloud: A Middleware Perspective*. Taylor & Francis, 2012. ISBN 9781439892992. Dostupné také z: https://books.google.cz/books?id=nIYKwoy1Z6oC.

22. FAJMON, Martin. *Velké srovnání cloudových úložišť: které je nejlepší? | mobilenet.cz* [https://mobilenet.cz/clanky/velke-srovnani-cloudovych-ulozist-ktere-je-nejlepsi-30300]. 2016-06. (Accessed on 12/18/2020).

23. KOĎOUSKOVÁ, Barbora. *Internet věcí (IoT): definice, příklady využití, produkty* [https://www.rascasone.com/cs/blog/iot-internet-veci-definice-produkty-historie]. 2021-04. (Accessed on 04/16/2021).

24. MICROSOFT. *Pricing—IoT Hub | Microsoft Azure* [https://azure.microsoft.com/en-us/pricing/details/iot-hub/]. 2021. (Accessed on 04/17/2021).

25. SYSTEM, Amon. *Co je to Microsoft Azure a kdy o něm přemýšlet? | by Amon System | Amon píše | Medium* [https://medium.com/amon-píše/co-je-to-microsoft-azure-a-kdy-o-něm-přemýšlet-f6721a949cc6]. 2017-06. (Accessed on 03/01/2021).

Příloha A

Příloha v IS EDISON

Seznam příloh

I. aplikace **ds18b20.py**

II. aplikace **main.py**